Combination Methods for Model-Checking of Infinite-State Systems

S. Ghilardi\textsuperscript{1}, E. Nicolini\textsuperscript{2}, S. Ranise\textsuperscript{2}, and D. Zucchelli\textsuperscript{1,2}

\textsuperscript{1}Università degli Studi di Milano
\textsuperscript{2}LORIA & INRIA-Lorraine

Berlin - July, 1 2007
Motivations

[Manna and Pnueli 1995]: First-Order Logic (FOL) + Linear time Temporal Logic (LTL) precisely state verification problems for the class of reactive systems;

- FOL: (possibly infinite) data structures used by a reactive system;
- LTL: dynamic behavior of a reactive system;
- LTL + FOL = interaction between data flow and control flow in infinite state systems.
Motivations

- [Manna and Pnueli 1995]: First-Order Logic (FOL) + Linear time Temporal Logic (LTL) precisely state verification problems for the class of reactive systems;
- FOL: (possibly infinite) data structures used by a reactive system;
- LTL: dynamic behavior of a reactive system;
- LTL + FOL = interaction between data flow and control flow in infinite state systems.
Motivations

- [Manna and Pnueli 1995]: First-Order Logic (FOL) + Linear time Temporal Logic (LTL) precisely state verification problems for the class of reactive systems;
- FOL: (possibly infinite) data structures used by a reactive system;
- LTL: dynamic behavior of a reactive system;
- LTL + FOL = interaction between data flow and control flow in infinite state systems.
Motivations

[Manna and Pnueli 1995]: First-Order Logic (FOL) + Linear time Temporal Logic (LTL) precisely state verification problems for the class of reactive systems;

- FOL: (possibly infinite) data structures used by a reactive system;
- LTL: dynamic behavior of a reactive system;
- LTL + FOL = interaction between data flow and control flow in infinite state systems.
The RoadMap

1. LTL-theories
2. Transition Systems
3. Main Result
The RoadMap

1. LTL-theories
2. Transition Systems
3. Main Result
The RoadMap

1. LTL-theories
2. Transition Systems
3. Main Result
The RoadMap

1. LTL-theories
2. Transition Systems
3. Main Result
Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1 LTL-theories

- First-order $\Sigma$-theory $T$;
- Temporal model: a sequence $M_1, M_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
- Symbols from $\Sigma_r \subseteq \Sigma$ are time independent, other symbols are time dependent.

2 Transition Systems

- The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3 Main Result

Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1. LTL-theories

   - First-order $\Sigma$-theory $T$;
   - Temporal model: a sequence $M_1, M_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
   - Symbols from $\Sigma_r \subseteq \Sigma$ are time independent, other symbols are time dependent.

2. Transition Systems

   - The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3. Main Result

Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1 LTL-theories

- First-order $\Sigma$-theory $T$;
- Temporal model: a sequence $M_1, M_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
- Symbols from $\Sigma_r \subseteq \Sigma$ are time independent, other symbols are time dependent.

2 Transition Systems

- The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3 Main Result

Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1. **LTL-theories**
   - First-order $\Sigma$-theory $T$;
   - Temporal model: a sequence $M_1, M_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
   - Symbols from $\Sigma_r \subseteq \Sigma$ are *time independent*, other symbols are *time dependent*.

2. **Transition Systems**
   - The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3. **Main Result**
Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1 LTL-theories

- First-order \( \Sigma \)-theory \( T \);
- Temporal model: a sequence \( \mathcal{M}_1, \mathcal{M}_2, \ldots \) of standard (first-order) models of \( T \) over the same carrier;
- Symbols from \( \Sigma_r \subseteq \Sigma \) are \textit{time independent}, other symbols are \textit{time dependent}.

2 Transition Systems

- The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3 Main Result

Introduction

Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1 LTL-theories

- First-order $\Sigma$-theory $T$;
- Temporal model: a sequence $\mathcal{M}_1, \mathcal{M}_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
- Symbols from $\Sigma_r \subseteq \Sigma$ are time independent, other symbols are time dependent.

2 Transition Systems

- The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3 Main Result

Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1. LTL-theories
   - First-order $\Sigma$-theory $T$;
   - Temporal model: a sequence $\mathcal{M}_1, \mathcal{M}_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
   - Symbols from $\Sigma_r \subseteq \Sigma$ are time independent, other symbols are time dependent.

2. Transition Systems
   - The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3. Main Result
Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1. LTL-theories
   - First-order $\Sigma$-theory $T$;
   - Temporal model: a sequence $\mathcal{M}_1, \mathcal{M}_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
   - Symbols from $\Sigma_r \subseteq \Sigma$ are time independent, other symbols are time dependent.

2. Transition Systems
   - The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3. Main Result
Safety Model-Checking Problem

The problem of checking if ‘bad states’ are reachable by a given transition system.

1. LTL-theories
   - First-order $\Sigma$-theory $T$;
   - Temporal model: a sequence $M_1, M_2, \ldots$ of standard (first-order) models of $T$ over the same carrier;
   - Symbols from $\Sigma_r \subseteq \Sigma$ are time independent, other symbols are time dependent.

2. Transition Systems
   - The initial/bad states and the transition relation are represented by first-order formulae, whose role is that of (non-deterministically) restricting the temporal evolution of the model.

3. Main Result
Introduction

The RoadMap

1. LTL-theories
**LTL-theory: Syntax**

**Definition (LTL-theory)**

An *LTL-theory* is a 5-tuple $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ where $\Sigma$ is a signature, $T$ is a $\Sigma$-theory (called the underlying theory of $\mathcal{T}$), $\Sigma_r$ is a subsignature of $\Sigma$, and $a, c$ are sets of free constants.

- $\Sigma_r$ is the *time-independent subsignature* of the LTL-theory;
- the constants $c$ (called *system parameters*) will be interpreted in a time-independent way;
- the constants $a$ (called *system variables*) will be interpreted in a time-dependent way.
LTL-theory: Syntax

Definition (LTL-theory)

An **LTL-theory** is a 5-tuple \( T = \langle \Sigma, T, \Sigma_r, a, c \rangle \) where \( \Sigma \) is a signature, \( T \) is a \( \Sigma \)-theory (called the underlying theory of \( T \)), \( \Sigma_r \) is a subsignature of \( \Sigma \), and \( a, c \) are sets of free constants.

- \( \Sigma_r \) is the *time-independent subsignature* of the LTL-theory;
- the constants \( c \) (called *system parameters*) will be interpreted in a time-independent way;
- the constants \( a \) (called *system variables*) will be interpreted in a time-dependent way.
LTL-theory: Syntax

**Definition (LTL-theory)**

An *LTL-theory* is a 5-tuple $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ where $\Sigma$ is a signature, $T$ is a $\Sigma$-theory (called the underlying theory of $\mathcal{T}$), $\Sigma_r$ is a subsignature of $\Sigma$, and $a, c$ are sets of free constants.

- $\Sigma_r$ is the *time-independent subsignature* of the LTL-theory;
- the constants $c$ (called *system parameters*) will be interpreted in a time-independent way;
- the constants $a$ (called *system variables*) will be interpreted in a time-dependent way.
### LTL-theory: Syntax

**Definition (LTL-theory)**

An **LTL-theory** is a 5-tuple \( \mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle \) where \( \Sigma \) is a signature, \( T \) is a \( \Sigma \)-theory (called the underlying theory of \( \mathcal{T} \)), \( \Sigma_r \) is a subsignature of \( \Sigma \), and \( a, c \) are sets of free constants.

- \( \Sigma_r \) is the **time-independent subsignature** of the LTL-theory;
- the constants \( c \) (called **system parameters**) will be interpreted in a time-independent way;
- the constants \( a \) (called **system variables**) will be interpreted in a time-dependent way.
LTL-theory: Semantic

Definition

An LTL(Σ, a, c)-structure $\mathcal{M} = \{\mathcal{M}_n = (M, I_n)\}_{n \in \mathbb{N}}$ is appropriate for an LTL-theory $T = \langle \Sigma, T, \Sigma_r, a, c \rangle$ iff we have

$$\mathcal{M}_n \models T, \quad I_n(f) = I_m(f), \quad I_n(P) = I_m(P), \quad I_n(c) = I_m(c).$$

for all $m, n \in \mathbb{N}$, for each function symbol $f \in \Sigma_r$, for each relational symbol $P \in \Sigma_r$, and for all constant $c \in c$. 
Locally finite compatible LTL-theories

Definition (Locally finite compatible LTL-theories)

An LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ is \textit{locally finite compatible} iff there is a universal and effectively locally finite $\Sigma_r$-theory $T_r$ such that $T$ is $T_r$-compatible and the constraint satisfiability problem for $T$ is decidable.

- $T_r$-compatibility and local finiteness requirements are the key ingredients to guarantee completeness and termination of our procedure;
- the (safety) model-checking problem we are going to introduce is related to a combination of infinite (partially renamed) copies of the theory $T$ sharing the common subtheory $T_r$. 
Locally finite compatible LTL-theories

Definition (Locally finite compatible LTL-theories)

An LTL-theory \( T = \langle \Sigma, T, \Sigma_r, a, c \rangle \) is locally finite compatible iff there is a universal and effectively locally finite \( \Sigma_r \)-theory \( T_r \) such that \( T \) is \( T_r \)-compatible and the constraint satisfiability problem for \( T \) is decidable.

- \( T_r \)-compatibility and local finiteness requirements are the key ingredients to guarantee completeness and termination of our procedure;
- the (safety) model-checking problem we are going to introduce is related to a combination of infinite (partially renamed) copies of the theory \( T \) sharing the common subtheory \( T_r \).
Locally finite compatible LTL-theories

**Definition (Locally finite compatible LTL-theories)**

An LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ is *locally finite compatible* iff there is a universal and effectively locally finite $\Sigma_r$-theory $\mathcal{T}_r$ such that $\mathcal{T}$ is $\mathcal{T}_r$-compatible and the constraint satisfiability problem for $\mathcal{T}$ is decidable.

- $\mathcal{T}_r$-compatibility and local finiteness requirements are the key ingredients to guarantee completeness and termination of our procedure;
- the (safety) model-checking problem we are going to introduce is related to a combination of infinite (partially renamed) copies of the theory $\mathcal{T}$ sharing the common subtheory $\mathcal{T}_r$. 

S. Ghilardi et al. (UNIMI & INRIA-Lorraine)  
Combination Methods for Model-Checking  
SMT07 8 / 26
Digression: Non-Disjoint Combination

1. The $T_r$-compatibility of a $\Sigma$-theory $T$ is a (quite technical) model-theoretic notion;
   - In practice (sufficient condition): $T$ includes a (universal) $\Sigma_r$-theory $T_r$ which admits quantifier elimination ($\Sigma_r \subseteq \Sigma$);
   - Notice that we do not need to have a characterization of $T_r$: the mere information of its existence is enough for our decision procedures to be sound and complete and to implement them;

2. A $\Sigma_r$-theory $T_r$ is (effectively) locally finite iff $\Sigma_r$ is finite and there exists a finite (and computable) set of terms that are “representative” modulo $T_r$-equivalence of the whole set of $\Sigma_r$-terms.
   - Examples: purely relational signature, orders, arithmetic modulo.
Digression: Non-Disjoint Combination

1. The $T_r$-compatibility of a $\Sigma$-theory $T$ is a (quite technical) model-theoretic notion;
   - In practice (sufficient condition): $T$ includes a (universal) $\Sigma_r$-theory $T_r$ which admits quantifier elimination ($\Sigma_r \subseteq \Sigma$);
   - Notice that we do not need to have a characterization of $T_r$: the mere information of its existence is enough for our decision procedures to be sound and complete and to implement them;

2. A $\Sigma_r$-theory $T_r$ is (effectively) locally finite iff $\Sigma_r$ is finite and there exists a finite (and computable) set of terms that are “representative” modulo $T_r$-equivalence of the whole set of $\Sigma_r$-terms.
   - Examples: purely relational signature, orders, arithmetic modulo.
Digression: Non-Disjoint Combination

1. The $T_r$-compatibility of a $\Sigma$-theory $T$ is a (quite technical) model-theoretic notion;
   - In practice (sufficient condition): $T$ includes a (universal) $\Sigma_r$-theory $T_r$ which admits quantifier elimination ($\Sigma_r \subseteq \Sigma$);
   - Notice that we do not need to have a characterization of $T_r$: the mere information of its existence is enough for our decision procedures to be sound and complete and to implement them;

2. A $\Sigma_r$-theory $T_r$ is (effectively) locally finite iff $\Sigma_r$ is finite and there exists a finite (and computable) set of terms that are “representative” modulo $T_r$-equivalence of the whole set of $\Sigma_r$-terms.
   - Examples: purely relational signature, orders, arithmetic modulo.
Digression: Non-Disjoint Combination

1. The $T_r$-compatibility of a $\Sigma$-theory $T$ is a (quite technical) model-theoretic notion;
   - In practice (sufficient condition): $T$ includes a (universal) $\Sigma_r$-theory $T_r$ which admits quantifier elimination ($\Sigma_r \subseteq \Sigma$);
   - Notice that we do not need to have a characterization of $T_r$: the mere information of its existence is enough for our decision procedures to be sound and complete and to implement them;

2. A $\Sigma_r$-theory $T_r$ is (effectively) locally finite iff $\Sigma_r$ is finite and there exists a finite (and computable) set of terms that are “representative” modulo $T_r$-equivalence of the whole set of $\Sigma_r$-terms.
   - Examples: purely relational signature, orders, arithmetic modulo.
Digression: Non-Disjoint Combination

1. The $T_r$-compatibility of a $\Sigma$-theory $T$ is a (quite technical) model-theoretic notion;
   - In practice (sufficient condition): $T$ includes a (universal) $\Sigma_r$-theory $T_r$ which admits quantifier elimination ($\Sigma_r \subseteq \Sigma$);
   - Notice that we do not need to have a characterization of $T_r$: the mere information of its existence is enough for our decision procedures to be sound and complete and to implement them;

2. A $\Sigma_r$-theory $T_r$ is \textit{(effectively) locally finite} iff $\Sigma_r$ is finite and there exists a finite (and computable) set of terms that are “representative” modulo $T_r$-equivalence of the whole set of $\Sigma_r$-terms.
   - Examples: purely relational signature, orders, arithmetic modulo.
The RoadMap

1. LTL-theories

2. Transition Systems
LTL-System Specifications: Syntax

Definition (LTL-System Specification)

An *LTL-system specification* is an LTL-theory \( \mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle \) (with finitely many system variables and parameters) endowed with a transition relation \( \delta(a^0, a^1) \) and with an initial state description \( \nu(a) \).

What is the transition relation \( \delta(a^0, a^1) \)?
LTL-System Specifications: Syntax

Definition (LTL-System Specification)

An LTL-system specification is an LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ (with finitely many system variables and parameters) endowed with a transition relation $\delta(a^0, a^1)$ and with an initial state description $\iota(a)$.

What is the transition relation $\delta(a^0, a^1)$?
Transition Relations: \((\Sigma \oplus \Sigma_r, \Sigma)\)-sentences

- We define the \textit{one-step signature} as
  \[
  \Sigma \oplus \Sigma_r := ((\Sigma \setminus \Sigma_r) \uplus (\Sigma \setminus \Sigma_r)) \cup \Sigma_r;
  \]

- \(\delta(a^0, a^1)\) is a \((\Sigma^{a,c} \oplus \Sigma^{c}_r, \Sigma^{a,c})\)-sentence, i.e. a sentence on the signature having two renamed occurrences of the time-dependent symbol \((r^0\text{ and } r^1\text{ for } r \in \Sigma^{a,c} \setminus \Sigma^{c}_r)\);

- A \((\Sigma^{a,c} \oplus \Sigma^{c}_r, \Sigma^{a,c})\)-structure for \(\delta(a^0, a^1)\) can be seen as \(M_0 \oplus \Sigma^{c}_r M_1\) where \(M_i\) are \(\Sigma^{a,c}\)-structures with the same \(\Sigma^{c}_r\)-reduct. (Combination!)
Transition Relations: \((\Sigma \oplus \Sigma_r \Sigma)\)-sentences

- We define the one-step signature as
  \[
  \Sigma \oplus \Sigma_r := ((\Sigma \setminus \Sigma_r) \cup (\Sigma \setminus \Sigma_r)) \cup \Sigma_r;
  \]

- \(\delta(\sigma^0, \sigma^1)\) is a \((\Sigma^{a,c} \oplus_{\Sigma_r^c} \Sigma^{a,c})\)-sentence, i.e. a sentence on the signature having two renamed occurrences of the time-dependent symbol \((r^0\text{ and } r^1 \text{ for } r \in \Sigma^{a,c} \setminus \Sigma^c_r)\);

- A \((\Sigma^{a,c} \oplus_{\Sigma_r^c} \Sigma^{a,c})\)-structure for \(\delta(\sigma^0, \sigma^1)\) can be seen as \(M_0 \oplus_{\Sigma_r^c} M_1\) where \(M_i\) are \(\Sigma^{a,c}\)-structures with the same \(\Sigma^c_r\)-reduct. (Combination!)
Transition Relations: \((\Sigma \oplus \Sigma_r \Sigma)\)-sentences

- We define the \textit{one-step signature} as
  \[
  \Sigma \oplus \Sigma := ((\Sigma \setminus \Sigma_r) \cup (\Sigma \setminus \Sigma_r)) \cup \Sigma_r;
  \]

- \(\delta(\alpha^0, \alpha^1)\) is a \((\Sigma^{a.c} \oplus_{\Sigma^c_r} \Sigma^{a.c})\)-sentence, i.e. a sentence on the signature having two renamed occurrences of the time-dependent symbol \((r^0\text{ and } r^1\text{ for } r \in \Sigma^{a.c} \setminus \Sigma^{c}_r)\);

- A \((\Sigma^{a.c} \oplus_{\Sigma^c_r} \Sigma^{a.c})\)-structure for \(\delta(\alpha^0, \alpha^1)\) can be seen as \(M_0 \oplus_{\Sigma^c_r} M_1\) where \(M_i\) are \(\Sigma^{a.c}\)-structures with the same \(\Sigma^c_r\)-reduct. (Combination!)
Transition Relations: \((\Sigma \oplus_{\Sigma_r} \Sigma)-sentences\)

- We define the \textit{one-step signature} as
  \[
  \Sigma \oplus_{\Sigma_r} \Sigma := ((\Sigma \setminus \Sigma_r) \cup (\Sigma \setminus \Sigma_r)) \cup \Sigma_r;
  \]

- \(\delta(a^0, a^1)\) is a \((\Sigma^{a,c} \oplus_{\Sigma_r} \Sigma^{a,c})\)-sentence, i.e. a sentence on the signature having two renamed occurrences of the time-dependent symbol \((r^0\) and \(r^1\) for \(r \in \Sigma^{a,c} \setminus \Sigma_r\));

- A \((\Sigma^{a,c} \oplus_{\Sigma_r} \Sigma^{a,c})\)-structure for \(\delta(a^0, a^1)\) can be seen as \(M_0 \oplus_{\Sigma_r} M_1\) where \(M_i\) are \(\Sigma^{a,c}\)-structures with the same \(\Sigma_r^c\)-reduct. \((\text{Combination!})\)
Safety Model-Checking

Transition Systems

LTL-System Specifications: Semantic

**Definition**

An LTL($\Sigma^{a,c}$)-structure $\mathcal{M} = \{\mathcal{M}_n = (M, I_n)\}_{n \in \mathbb{N}}$ is a run for an LTL-system specification $(\mathcal{T}, \delta, \iota)$ iff it is appropriate for $\mathcal{T}$ and

1. $\mathcal{M}_0 \models \iota(a)$
2. $\mathcal{M}_n \bigoplus_{\Sigma^c} \mathcal{M}_{n+1} \models \delta(a^0, a^1)$, for every $n \geq 0$. 

S. Ghilardi et al. (UNIMI & INRIA-Lorraine)  Combination Methods for Model-Checking
LTL-System Specifications: Semantic

Definition

An LTL(Σᵃ,ᶜ)-structure \( \mathcal{M} = \{ \mathcal{M}_n = (\mathcal{M}, I_n) \}_{n \in \mathbb{N}} \) is a run for an LTL-system specification \((\mathcal{T}, \delta, \iota)\) iff it is appropriate for \(\mathcal{T}\) and

1. \( \mathcal{M}_0 \models \iota(a) \)
2. \( \mathcal{M}_n \oplus_{\Sigma_T} \mathcal{M}_{n+1} \models \delta(a^0, a^1), \) for every \( n \geq 0. \)
LTL-System Specifications: Semantic

Definition

An LTL($\Sigma^a,c$)-structure $\mathcal{M} = \{\mathcal{M}_n = (M, I_n)\}_{n \in \mathbb{N}}$ is a run for an LTL-system specification $(T, \delta, \iota)$ iff it is appropriate for $T$ and

1. $\mathcal{M}_0 \models \iota(a)$
2. $\mathcal{M}_n \oplus_{\Sigma_T} \mathcal{M}_{n+1} \models \delta(a^0, a^1)$, for every $n \geq 0$. 

S. Ghilardi et al. (UNIMI & INRIA-Lorraine) Combination Methods for Model-Checking
The RoadMap

1. LTL-theories
2. Transition Systems
3. Main Result
Statement of the Problem

Definition (Safety Model-Checking Problem)

The safety model-checking problem for the system specification \((\mathcal{T}, \delta, \iota)\) is the following: decide whether there is a run \(\mathcal{M} = \{\mathcal{M}_n = (M, I_n)\}_{n \in \mathbb{N}}\) for \((\mathcal{T}, \delta, \iota)\) such that \(\mathcal{M}_n \models \nu\) for some \(n \in \mathbb{N}\). The system specification \((\mathcal{T}, \delta, \iota)\) is safe for \(\nu\) iff the safety model-checking problem for \(\nu\) has a negative solution.

- Here \(\nu\) is a \(\Sigma_{a,c}\)-sentence describes the set of unsafe states;
- When \(\delta, \iota\) and \(\nu\) are ground sentences, we talk of ground safety model-checking problem.

Main result: the ground safety model-checking problem for locally finite compatible LTL-theories is decidable!
Statement of the Problem

Definition (Safety Model-Checking Problem)

The \textit{safety model-checking problem} for the system specification \((T, \delta, \iota)\) is the following: decide whether there is a run \(M = \{M_n = (M, I_n)\}_{n \in \mathbb{N}}\) for \((T, \delta, \iota)\) such that \(M_n \models \nu\) for some \(n \in \mathbb{N}\). The system specification \((T, \delta, \iota)\) is \textit{safe for} \(\nu\) iff the safety model-checking problem for \(\nu\) has a negative solution.

- Here \(\nu\) is a \(\Sigma^{a.c.}\)-sentence describes the set of unsafe states;
- When \(\delta, \iota\) and \(\nu\) are ground sentences, we talk of \textit{ground safety model-checking problem}.

Main result: the \textit{ground safety model-checking problem} for locally finite compatible LTL-theories is decidable!
Statement of the Problem

Definition (Safety Model-Checking Problem)

The safety model-checking problem for the system specification \((\mathcal{T}, \delta, \iota)\) is the following: decide whether there is a run \(\mathcal{M} = \{\mathcal{M}_n = (M, I_n)\}_{n \in \mathbb{N}}\) for \((\mathcal{T}, \delta, \iota)\) such that \(\mathcal{M}_n \models \upsilon\) for some \(n \in \mathbb{N}\). The system specification \((\mathcal{T}, \delta, \iota)\) is safe for \(\upsilon\) iff the safety model-checking problem for \(\upsilon\) has a negative solution.

- Here \(\upsilon\) is a \(\Sigma^{\mathbf{a}, \mathbf{c}}\)-sentence describes the set of unsafe states;
- When \(\delta, \iota\) and \(\upsilon\) are ground sentences, we talk of ground safety model-checking problem.

Main result: the ground safety model-checking problem for locally finite compatible LTL-theories is decidable!
Statement of the Problem

Definition (Safety Model-Checking Problem)

The safety model-checking problem for the system specification \((\mathcal{T}, \delta, \iota)\) is the following: decide whether there is a run \(\mathcal{M} = \{\mathcal{M}_n = (M, I_n)\}_{n \in \mathbb{N}}\) for \((\mathcal{T}, \delta, \iota)\) such that \(\mathcal{M}_n \models \nu\) for some \(n \in \mathbb{N}\). The system specification \((\mathcal{T}, \delta, \iota)\) is safe for \(\nu\) iff the safety model-checking problem for \(\nu\) has a negative solution.

- Here \(\nu\) is a \(\Sigma^{a,c}\)-sentence describes the set of unsafe states;
- When \(\delta\), \(\iota\) and \(\nu\) are ground sentences, we talk of ground safety model-checking problem.

Main result: the ground safety model-checking problem for locally finite compatible LTL-theories is decidable!
Safety Model-Checking

Statement of the Problem

Definition (Safety Model-Checking Problem)

The safety model-checking problem for the system specification \((\mathcal{T}, \delta, \iota)\) is the following: decide whether there is a run \(\mathcal{M} = \{\mathcal{M}_n = (M, I_n)\}_{n\in\mathbb{N}}\) for \((\mathcal{T}, \delta, \iota)\) such that \(\mathcal{M}_n \models \nu\) for some \(n \in \mathbb{N}\). The system specification \((\mathcal{T}, \delta, \iota)\) is safe for \(\nu\) iff the safety model-checking problem for \(\nu\) has a negative solution.

- Here \(\nu\) is a \(\Sigma^{a,c}\)-sentence describes the set of unsafe states;
- When \(\delta, \iota\) and \(\nu\) are ground sentences, we talk of ground safety model-checking problem.

Main result: the ground safety model-checking problem for locally finite compatible LTL-theories is decidable!
Main Definition

Definition (Safety Graph)

The safety graph associated to the LTL-system specification $(\mathcal{T}, \delta, \iota)$ based on the locally finite compatible LTL-theory $\mathcal{T}$ is the directed graph defined as follows:

- the nodes are the pairs $(V, G)$ where $V$ is a $\tilde{\delta}$-assignment and $G$ is a transition $\Sigma_r$-guessing;
- there is an edge $(V, G) \rightarrow (W, H)$ iff the ground sentence

$$G(a^0, a^1, d^0) \land V^r(a^0, a^1, d^0) \land W^l(a^1, a^2, d^1) \land H(a^1, a^2, d^1)$$

is $T$-satisfiable.
Main Definition

Definition (Safety Graph)

The safety graph associated to the LTL-system specification \((\mathcal{T}, \delta, \iota)\) based on the locally finite compatible LTL-theory \(\mathcal{T}\) is the directed graph defined as follows:

- the nodes are the pairs \((V, G)\) where \(V\) is a \(\tilde{\delta}\)-assignment and \(G\) is a transition \(\Sigma_r\)-guessing;
- there is an edge \((V, G) \to (W, H)\) iff the ground sentence

\[
G(a^0, a^1, d^0) \land V^r(a^0, a^1, d^0) \land W'(a^1, a^2, d^1) \land H(a^1, a^2, d^1)
\]

is \(\mathcal{T}\)-satisfiable.
Main Definition

Definition (Safety Graph)

The safety graph associated to the LTL-system specification \((\mathcal{T}, \delta, \iota)\) based on the locally finite compatible LTL-theory \(\mathcal{T}\) is the directed graph defined as follows:

- the nodes are the pairs \((V, G)\) where \(V\) is a \(\tilde{\delta}\)-assignment and \(G\) is a transition \(\Sigma_r\)-guessing;
- there is an edge \((V, G) \rightarrow (W, H)\) iff the ground sentence

\[
G(a^0, a^1, d^0) \land V'(a^0, a^1, d^0) \land W'(a^1, a^2, d^1) \land H(a^1, a^2, d^1)
\]

is \(T\)-satisfiable.
A Technical Concept

Our main definition of safety graph relies on the following

Definition (Purely Left/Right Sentence)

A ground \((\Sigma^{a,c} \oplus \Sigma^c \Sigma^{a,c})\)-sentence \(\delta\) is said to be purely left (purely right) iff for each symbol \(r \in \Sigma \setminus \Sigma_r\), we have that \(r^1\) (\(r^0\), resp.) does not occur in \(\delta\).

- Each assignment to the atoms of the purification \(\tilde{\delta}\) of the transition relation \(\delta\) can be seen as a conjunction of purely left and purely right literals.
A Technical Concept

Our main definition of safety graph relies on the following

**Definition (Purely Left/Right Sentence)**

A ground \((\Sigma^a \cdot \Sigma^c \oplus \Sigma^c \cdot \Sigma^a)\)-sentence \(\delta\) is said to be purely left (purely right) iff for each symbol \(r \in \Sigma \setminus \Sigma_r\), we have that \(r^1\) (\(r^0\), resp.) does not occur in \(\delta\).

Each assignment to the atoms of the purification \(\tilde{\delta}\) of the transition relation \(\delta\) can be seen as a conjunction of purely left and purely right literals.
**Main Definition**

**Definition (Safety Graph)**

The *safety graph* associated to the LTL-system specification $(T, \delta, \iota)$ based on the locally finite compatible LTL-theory $T$ is the directed graph defined as follows:

- the nodes are the pairs $(V, G)$ where $V$ is a $\tilde{\delta}$-assignment and $G$ is a transition $\Sigma_r$-guessing;
- there is an edge $(V, G) \rightarrow (W, H)$ iff the ground sentence

$$G(a^0, a^1, d^0) \land V^r(a^0, a^1, d^0) \land W^l(a^1, a^2, d^1) \land H(a^1, a^2, d^1)$$

is $T$-satisfiable.

- **Initial nodes**: nodes $(V, G)$ such that $\iota(a^0) \land V^l(a^0, a^1, d^0) \land G(a^0, a^1, d^0)$ is $T$-satisfiable;
- **Terminal nodes**: nodes $(V, G)$ such that $V^r(a^0, a^1, d^0) \land \iota(a^1) \land G(a^0, a^1, d^0)$ is $T$-satisfiable.
Main Definition

Definition (Safety Graph)

The safety graph associated to the LTL-system specification \((T, \delta, \iota)\) based on the locally finite compatible LTL-theory \(T\) is the directed graph defined as follows:

- the nodes are the pairs \((V, G)\) where \(V\) is a \(\tilde{\delta}\)-assignment and \(G\) is a transition \(\Sigma_r\)-guessing;
- there is an edge \((V, G) \rightarrow (W, H)\) iff the ground sentence
  \[
  G(a^0, a^1, d^0) \land V^r(a^0, a^1, d^0) \land W^l(a^1, a^2, d^1) \land H(a^1, a^2, d^1)
  \]
  is \(T\)-satisfiable.

initial nodes nodes \((V, G)\) such that \(\iota(a^0) \land V^l(a^0, a^1, d^0) \land G(a^0, a^1, d^0)\) is \(T\)-satisfiable;

terminal nodes nodes \((V, G)\) such that \(V^r(a^0, a^1, d^0) \land \nu(a^1) \land G(a^0, a^1, d^0)\) is \(T\)-satisfiable.
Main Definition

Definition (Safety Graph)

The safety graph associated to the LTL-system specification \((T, \delta, \iota)\) based on the locally finite compatible LTL-theory \(T\) is the directed graph defined as follows:

- the nodes are the pairs \((V, G)\) where \(V\) is a \(\tilde{\delta}\)-assignment and \(G\) is a transition \(\Sigma_r\)-guessing;
- there is an edge \((V, G) \rightarrow (W, H)\) iff the ground sentence

\[
G(a^0, a^1, d^0) \land V^r(a^0, a^1, d^0) \land W^l(a^1, a^2, d^1) \land H(a^1, a^2, d^1)
\]

is \(T\)-satisfiable.

- initial nodes: nodes \((V, G)\) such that \(\iota(a^0) \land V^l(a^0, a^1, d^0) \land G(a^0, a^1, d^0)\) is \(T\)-satisfiable;
- terminal nodes: nodes \((V, G)\) such that \(V^r(a^0, a^1, d^0) \land \upsilon(a^1) \land G(a^0, a^1, d^0)\) is \(T\)-satisfiable.
Main Result

Proposition

The system is unsafe iff either $\nu(a) \land \nu(a) \text{ is } T\text{-satisfiable}$ or there is a path in the safety graph from an initial to a terminal node.

Proof (Sketch)

A bad run of length $n + 1$ exists iff, for some $\tilde{\delta}$-assignments $V_1, \ldots, V_{n+1}$, the ground $(\bigoplus_{c}^{n+2} \Sigma^{a,c})$-sentence

$$\nu(a^0) \land \bigwedge_{i=0}^{n} (V_{i+1}^{f}(a^i, a^{i+1}, d^i) \land V_{i+1}^{f}(a^i, a^{i+1}, d^i)) \land \nu(a^{n+1})$$

(1)

is $\bigoplus_{c}^{n+2} T\text{-satisfiable}$. By contradiction, assume there is a path from an initial to a terminal node and the system is safe. Repeatedly, compute $\Sigma_r$-ground interpolants of (1) between $T$ and $\bigoplus_{\Sigma_r}^{j} T$, for $j = n + 1, \ldots, 1$ (an argument based on $T_r$-compatibility guarantees they exist). This yields the $T\text{-unsatisfiability}$ of the final node (formula) in the graph; contradiction.
Main Result

Proposition

The system is unsafe iff either $\nu(a) \land \nu(a)$ is $T$-satisfiable or there is a path in the safety graph from an initial to a terminal node.

Proof (Sketch)

A bad run of length $n+1$ exists iff, for some $\tilde{\delta}$-assignments $V_1, \ldots, V_{n+1}$, the ground $(\bigoplus_{\Sigma_r^{c}} \Sigma_r^{a,c})$-sentence

$$\nu(a^0) \land \bigwedge_{i=0}^{n} (V_{i+1}(a^i, a^{i+1}, d^i) \land V_{i+1}(a^i, a^{i+1}, d^i)) \land \nu(a^{n+1})$$  \hspace{1cm} (1)

is $\bigoplus_{\Sigma_r^{c}} T$-satisfiable. By contradiction, assume there is a path from an initial to a terminal node and the system is safe. Repeatedly, compute $\Sigma_r$-ground interpolants of (1) between $T$ and $\bigoplus_{\Sigma_r^{c}} T$, for $j = n+1, \ldots, 1$ (an argument based on $T_r$-compatibility guarantees they exist). This yields the $T$-unsatisfiability of the final node (formula) in the graph; contradiction.
Main Result

Proposition

The system is unsafe iff either $\iota(a) \land \nu(a)$ is $T$-satisfiable or there is a path in the safety graph from an initial to a terminal node.

Proof (Sketch)

A bad run of length $n + 1$ exists iff, for some $\tilde{\delta}$-assignments $V_1, \ldots, V_{n+1}$, the ground $(\bigoplus_{\Sigma_r}^{n+2} \Sigma^{a,c})$-sentence

$$\iota(a^0) \land \bigwedge_{i=0}^{n} (V_{i+1}^t(a^i, a^{i+1}, d^i) \land V_{i+1}^r(a^i, a^{i+1}, d^i)) \land \nu(a^{n+1}) \quad (1)$$

is $\bigoplus_{\Sigma_r}^{n+2} T$-satisfiable. By contradiction, assume there is a path from an initial to a terminal node and the system is safe. Repeatedly, compute $\Sigma_r$-ground interpolants of (1) between $T$ and $\bigoplus_{\Sigma_r} T$, for $j = n + 1, \ldots, 1$ (an argument based on $T_r$-compatibility guarantees they exist). This yields the $T$-unsatisfiability of the final node (formula) in the graph; contradiction.
Main Result

Proposition

The system is unsafe iff either $\iota(a) \land \nu(a)$ is $T$-satisfiable or there is a path in the safety graph from an initial to a terminal node.

Proof (Sketch)

A bad run of length $n + 1$ exists iff, for some $\tilde{\delta}$-assignments $V_1, \ldots, V_{n+1}$, the ground $(\bigoplus_{\Sigma_r}^{n+2} \Sigma_c^{a.c})$-sentence

$$\iota(a^0) \land \bigwedge_{i=0}^{n} (V'_{i+1}(a^i, a^{i+1}, d^i) \land V^r_{i+1}(a^i, a^{i+1}, d^i)) \land \nu(a^{n+1})$$

(1)

is $\bigoplus_{\Sigma_r}^{n+2} T$-satisfiable. By contradiction, assume there is a path from an initial to a terminal node and the system is safe. Repeatedly, compute $\Sigma_r$-ground interpolants of (1) between $T$ and $\bigoplus_{\Sigma_r}^j T$, for $j = n + 1, \ldots, 1$ (an argument based on $T_r$-compatibility guarantees they exist). This yields the $T$-unsatisfiability of the final node (formula) in the graph; contradiction.
Main Result

Proposition

The system is unsafe iff either $\nu(\mathbf{a}) \land \nu(\mathbf{a})$ is $T$-satisfiable or there is a path in the safety graph from an initial to a terminal node.

Proof (Sketch)

A bad run of length $n + 1$ exists iff, for some $\tilde{\delta}$-assignments $V_1, \ldots, V_{n+1}$, the ground $(\bigoplus_{c}^{n+2} \Sigma^{a,c})$-sentence

$$\nu(\mathbf{a}^0) \land \bigwedge_{i=0}^{n} (V_{i+1}^l(\mathbf{a}^i, \mathbf{a}^{i+1}, \mathbf{d}^i) \land V_{i+1}^r(\mathbf{a}^i, \mathbf{a}^{i+1}, \mathbf{d}^i)) \land \nu(\mathbf{a}^{n+1})$$

(1)

is $\bigoplus_{c}^{n+2} T$-satisfiable. By contradiction, assume there is a path from an initial to a terminal node and the system is safe. Repeatedly, compute $\Sigma_r$-ground interpolants of (1) between $T$ and $\bigoplus_{c}^{j} T$, for $j = n + 1, \ldots, 1$ (an argument based on $T_r$-compatibility guarantees they exist). This yields the $T$-unsatisfiability of the final node (formula) in the graph; contradiction.
Behind the Proof

Main ideas:

- ‘splitting’ the transition $\delta(\vec{a}^0, \vec{a}^1)$ into its left and right assignments allows to obtain (by Craig Interpolation Lemma) interpolants over the time-independent subsignature $\Sigma_r$;
- $T_r$-compatibility allows to conclude that the interpolants are ground;
- effective local finiteness allows to trade guessings for interpolants.
Behind the Proof

Main ideas:

- ‘splitting’ the transition $\delta(\bar{a}^0, \bar{a}^1)$ into its left and right assignments allows to obtain (by Craig Interpolation Lemma) interpolants over the time-independent subsignature $\Sigma_r$;
- $T_r$-compatibility allows to conclude that the interpolants are ground;
- effective local finiteness allows to trade guessings for interpolants.
Behind the Proof

Main ideas:

- ‘splitting’ the transition $\delta(\bar{a}^0, \bar{a}^1)$ into its left and right assignments allows to obtain (by Craig Interpolation Lemma) interpolants over the time-independent subsignature $\Sigma_r$;
- $T_r$-compatibility allows to conclude that the interpolants are ground;
- effective local finiteness allows to trade guessings for interpolants.
The RoadMap

1. LTL-theories
2. Transition Systems
3. Main Result
   - An Example
An Example

Consider a water level controller [Sofronie-Stokkermans 2006] such that:

- changes in the water level by $in(flow)/out(flow)$ depend on the water level $l$ and on the time instant;
- if $l \geq l_{\text{alarm}}$ at a given state (where $l_{\text{alarm}}$ is a fixed value), then a valve is opened and, at the next observable instant, $l' = in(out(l))$;
- if $l < l_{\text{alarm}}$ then the valve is closed and, at the next observable instant, $l' = in(l)$.

We want to check that, if in the initial state $l < l_{\text{alarm}}$, then it will never happen that $l_{\text{overflow}} < l$, for a fixed value $l_{\text{alarm}} < l_{\text{overflow}}$. 
An Example

Consider a water level controller [Sofronie-Stokkermans 2006] such that:

- changes in the water level by $\text{in}(\text{flow})/\text{out}(\text{flow})$ depend on the water level $l$ and on the time instant;
- if $l \geq l_{\text{alarm}}$ at a given state (where $l_{\text{alarm}}$ is a fixed value), then a valve is opened and, at the next observable instant, $l' = \text{in}(\text{out}(l))$;
- if $l < l_{\text{alarm}}$ then the valve is closed and, at the next observable instant, $l' = \text{in}(l)$.

We want to check that, if in the initial state $l < l_{\text{alarm}}$, then it will never happen that $l_{\text{overflow}} < l$, for a fixed value $l_{\text{alarm}} < l_{\text{overflow}}$. 
An Example

Consider a water level controller [Sofronie-Stokkermans 2006] such that:

- changes in the water level by $in(flow)/out(flow)$ depend on the water level $l$ and on the time instant;
- if $l \geq l_{\text{alarm}}$ at a given state (where $l_{\text{alarm}}$ is a fixed value), then a valve is opened and, at the next observable instant, $l' = in(out(l))$;
- if $l < l_{\text{alarm}}$ then the valve is closed and, at the next observable instant, $l' = in(l)$.

We want to check that, if in the initial state $l < l_{\text{alarm}}$, then it will never happen that $l_{\text{overflow}} < l$, for a fixed value $l_{\text{alarm}} < l_{\text{overflow}}$. 
An Example

Consider a water level controller [Sofronie-Stokkermans 2006] such that:

- changes in the water level by $in(flow)/out(flow)$ depend on the water level $l$ and on the time instant;
- if $l \geq l_{\text{alarm}}$ at a given state (where $l_{\text{alarm}}$ is a fixed value), then a valve is opened and, at the next observable instant, $l' = in(out(l))$;
- if $l < l_{\text{alarm}}$ then the valve is closed and, at the next observable instant, $l' = in(l)$.

We want to check that, if in the initial state $l < l_{\text{alarm}}$, then it will never happen that $l_{\text{overflow}} < l$, for a fixed value $l_{\text{alarm}} < l_{\text{overflow}}$. 
An Example

The associated LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ is the following:

- $a := \{l\}$ and $c := \emptyset$;
- $\Sigma_r = \{l_{\text{alarm}}, l_{\text{overflow}}, <\}$, ($l_{\text{alarm}}, l_{\text{overflow}}$ are constants);
- $\Sigma := \Sigma_r \cup \{\text{in}, \text{out}\}$;
- $T_r$ is the theory of dense linear orders without endpoints enriched with $l_{\text{alarm}} < l_{\text{overflow}}$;
- the theory $T$ is the following

$$T := T_r \cup \left\{ \forall x (x < l_{\text{alarm}} \rightarrow \text{in}(x) < l_{\text{overflow}}), \forall x (x < l_{\text{overflow}} \rightarrow \text{out}(x) < l_{\text{alarm}}) \right\}$$

It is possible to show that $\mathcal{T}$ is a locally finitely compatible LTL-theory.
An Example

The associated LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ is the following:

- $a := \{l\}$ and $c := \emptyset$;
- $\Sigma_r = \{l_{\text{alarm}}, l_{\text{overflow}}, <\}$, ($l_{\text{alarm}}, l_{\text{overflow}}$ are constants);
- $\Sigma := \Sigma_r \cup \{in, out\}$;
- $T_r$ is the theory of dense linear orders without endpoints enriched with $l_{\text{alarm}} < l_{\text{overflow}}$;
- the theory $T$ is the following

$$T := T_r \cup \left\{ \forall x (x < l_{\text{alarm}} \rightarrow in(x) < l_{\text{overflow}}), \forall x (x < l_{\text{overflow}} \rightarrow out(x) < l_{\text{alarm}}) \right\}$$

It is possible to show that $\mathcal{T}$ is a locally finitely compatible LTL-theory.
An Example

The associated LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ is the following:

- $a := \{l\}$ and $c := \emptyset$;
- $\Sigma_r = \{l_{\text{alarm}}, l_{\text{overflow}}, <\}$, ($l_{\text{alarm}}, l_{\text{overflow}}$ are constants);
- $\Sigma := \Sigma_r \cup \{\text{in, out}\}$;
- $T_r$ is the theory of dense linear orders without endpoints enriched with $l_{\text{alarm}} < l_{\text{overflow}}$;
- the theory $T$ is the following

$$T := T_r \cup \left\{ \forall x \left( x < l_{\text{alarm}} \rightarrow \text{in}(x) < l_{\text{overflow}} \right), \forall x \left( x < l_{\text{overflow}} \rightarrow \text{out}(x) < l_{\text{alarm}} \right) \right\}$$

It is possible to show that $\mathcal{T}$ is a locally finitely compatible LTL-theory.
An Example

The associated LTL-theory \( \mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle \) is the following:

- \( a := \{ l \} \) and \( c := \emptyset \);
- \( \Sigma_r = \{ l_{\text{alarm}}, l_{\text{overflow}}, < \} \), (\( l_{\text{alarm}}, l_{\text{overflow}} \) are constants);
- \( \Sigma := \Sigma_r \cup \{ \text{in}, \text{out} \} \);
- \( T_r \) is the theory of dense linear orders without endpoints enriched with \( l_{\text{alarm}} < l_{\text{overflow}} \);
- the theory \( T \) is the following

\[
T := T_r \cup \left\{ \begin{array}{l}
\forall x (x < l_{\text{alarm}} \rightarrow \text{in}(x) < l_{\text{overflow}}), \\
\forall x (x < l_{\text{overflow}} \rightarrow \text{out}(x) < l_{\text{alarm}})
\end{array} \right\}
\]

It is possible to show that \( \mathcal{T} \) is a locally finitely compatible LTL-theory.
An Example

The associated LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ is the following:

- $a := \{ l \}$ and $c := \emptyset$;
- $\Sigma_r = \{ l_{\text{alarm}}, l_{\text{overflow}}, < \}$, ($l_{\text{alarm}}, l_{\text{overflow}}$ are constants);
- $\Sigma := \Sigma_r \cup \{ in, out \}$;
- $T_r$ is the theory of dense linear orders without endpoints enriched with $l_{\text{alarm}} < l_{\text{overflow}}$;
- the theory $T$ is the following

$$T := T_r \cup \left\{ \forall x \left( x < l_{\text{alarm}} \rightarrow in(x) < l_{\text{overflow}} \right), \forall x \left( x < l_{\text{overflow}} \rightarrow out(x) < l_{\text{alarm}} \right) \right\}$$

It is possible to show that $\mathcal{T}$ is a locally finitely compatible LTL-theory.
An Example

The associated LTL-theory $\mathcal{T} = \langle \Sigma, T, \Sigma_r, a, c \rangle$ is the following:

- $a := \{l\}$ and $c := \emptyset$;
- $\Sigma_r = \{l_{\text{alarm}}, l_{\text{overflow}}, <\}$, ($l_{\text{alarm}}$, $l_{\text{overflow}}$ are constants);
- $\Sigma := \Sigma_r \cup \{\text{in}, \text{out}\}$;
- $T_r$ is the theory of dense linear orders without endpoints enriched with $l_{\text{alarm}} < l_{\text{overflow}}$;
- the theory $T$ is the following

$$T := T_r \cup \left\{ \forall x \ (x < l_{\text{alarm}} \rightarrow \text{in}(x) < l_{\text{overflow}}), \ \forall x \ (x < l_{\text{overflow}} \rightarrow \text{out}(x) < l_{\text{alarm}}) \right\}$$

It is possible to show that $\mathcal{T}$ is a locally finitely compatible LTL-theory.
An Example

We consider now the LTL-system specification \((\mathcal{T}, \delta, \nu)\) where

- \(\nu := l < l_{\text{alarm}}\);
- \(\delta := (l_{\text{alarm}} \leq l^0 \rightarrow l^1 = in^0(out^0(l^0))) \land (l^0 < l_{\text{alarm}} \rightarrow l^1 = in^0(l^0))\);
- \(\delta\) is a purely left \(\Sigma^a \oplus \Sigma^r, \Sigma^a\)-formula;
- \(\nu := l_{\text{overflow}} < l\) is the bad state condition.
An Example

We consider now the LTL-system specification ($\mathcal{I}, \delta, \nu$) where

- $\nu := l < l_{\text{alarm}}$;
- $\delta := (l_{\text{alarm}} \leq l^0 \rightarrow l^1 = \text{in}^0(\text{out}^0(l^0))) \land (l^0 < l_{\text{alarm}} \rightarrow l^1 = \text{in}^0(l^0))$;
- $\delta$ is a purely left ($\Sigma_a \oplus \Sigma_r, \Sigma_a$)-formula;
- $\nu := l_{\text{overflow}} < l$ is the bad state condition.
An Example

We consider now the LTL-system specification \((\mathcal{T}, \delta, \nu)\) where

- \(\nu := l < l_{\text{alarm}}\);
- \(\delta := (l_{\text{alarm}} \leq l^0 \rightarrow l^1 = in^0(out^0(l^0))) \land (l^0 < l_{\text{alarm}} \rightarrow l^1 = in^0(l^0))\);
- \(\delta\) is a purely left \((\Sigma^a \oplus \Sigma_r, \Sigma^a)\)-formula;
- \(\nu := l_{\text{overflow}} < l\) is the bad state condition.
An Example

We consider now the LTL-system specification \((\mathcal{T}, \delta, \iota)\) where

- \(\iota := l < l_{\text{alarm}}\);
- \(\delta := (l_{\text{alarm}} \leq l^0 \rightarrow l^1 = in^0(out^0(l^0))) \land (l^0 < l_{\text{alarm}} \rightarrow l^1 = in^0(l^0))\);
- \(\delta\) is a purely left \((\Sigma^a \oplus \Sigma_r, \Sigma^a)\)-formula;
- \(\nu := l_{\text{overflow}} < l\) is the bad state condition.
An Example

Building the associated safety graph:

- only 50 nodes are $T$-satisfiable (i.e. the nodes $(V, G)$ such that $V \land G$ is $T$-satisfiable);
- considering just the paths starting from the initial nodes
  - only 26 nodes are forward reachable;
- considering just the paths ending in the terminal nodes
  - only 12 nodes are backward reachable.

The combination of a decision procedure for $T$ with a SAT-solver (for enumerating the $\tilde{\delta}$-assignments) makes the problem easy to be automatically solved!
An Example

Building the associated safety graph:

- only 50 nodes are $T$-satisfiable (i.e. the nodes $(V, G)$ such that $V \land G$ is $T$-satisfiable);
- considering just the paths starting from the initial nodes
  - only 26 nodes are forward reachable;
- considering just the paths ending in the terminal nodes
  - only 12 nodes are backward reachable.

The combination of a decision procedure for $T$ with a SAT-solver (for enumerating the $\tilde{\delta}$-assignments) makes the problem easy to be automatically solved!
An Example

Building the associated safety graph:

- only 50 nodes are $T$-satisfiable (i.e. the nodes $(V, G)$ such that $V \land G$ is $T$-satisfiable);
- considering just the paths starting from the initial nodes
  - only 26 nodes are forward reachable;
- considering just the paths ending in the terminal nodes
  - only 12 nodes are backward reachable.

The combination of a decision procedure for $T$ with a SAT-solver (for enumerating the $\tilde{\delta}$-assignments) makes the problem easy to be automatically solved!
An Example

Building the associated safety graph:

- only 50 nodes are $T$-satisfiable (i.e. the nodes $(V, G)$ such that $V \land G$ is $T$-satisfiable);
- considering just the paths starting from the initial nodes
  - only 26 nodes are forward reachable;
- considering just the paths ending in the terminal nodes
  - only 12 nodes are backward reachable.

The combination of a decision procedure for $T$ with a SAT-solver (for enumerating the $\tilde{\delta}$-assignments) makes the problem easy to be automatically solved!
Building the associated safety graph:

- only 50 nodes are $T$-satisfiable (i.e. the nodes $(V, G)$ such that $V \land G$ is $T$-satisfiable);
- considering just the paths starting from the initial nodes
  - only 26 nodes are forward reachable;
- considering just the paths ending in the terminal nodes
  - only 12 nodes are backward reachable.

The combination of a decision procedure for $T$ with a SAT-solver (for enumerating the $\tilde{\delta}$-assignments) makes the problem easy to be automatically solved!
An Example

Building the associated safety graph:

- only 50 nodes are $T$-satisfiable (i.e. the nodes $(V, G)$ such that $V \land G$ is $T$-satisfiable);
- considering just the paths starting from the initial nodes
  - only 26 nodes are forward reachable;
- considering just the paths ending in the terminal nodes
  - only 12 nodes are backward reachable.

The combination of a decision procedure for $T$ with a SAT-solver (for enumerating the $\tilde{\delta}$-assignments) makes the problem easy to be automatically solved!
Conclusions and Future Work

- We have given the decidability of the restriction to safety properties of the model-checking problem modulo locally finite and compatible theories;
- Three main lines of future work:
  - how to exploit SMT solvers to solve model-checking problems (i.e., find suitable heuristics to efficiently explore the safety graph);
  - find decidability results for model checking of arbitrary temporal properties and modulo richer background theories [Demri et al. 2006];
  - handle universally quantified transition relations and initial state descriptions.
Conclusions and Future Work

- We have given the decidability of the restriction to safety properties of the model-checking problem modulo locally finite and compatible theories;
- Three main lines of future work:
  1. how to exploit SMT solvers to solve model-checking problems (i.e., find suitable heuristics to efficiently explore the safety graph);
  2. find decidability results for model checking of arbitrary temporal properties and modulo richer background theories [Demri et al. 2006];
  3. handle universally quantified transition relations and initial state descriptions.
Conclusions and Future Work

- We have given the decidability of the restriction to safety properties of the model-checking problem modulo locally finite and compatible theories;
- Three main lines of future work:
  1. how to exploit SMT solvers to solve model-checking problems (i.e., find suitable heuristics to efficiently explore the safety graph);
  2. find decidability results for model checking of arbitrary temporal properties and modulo richer background theories [Demri et al. 2006];
  3. handle universally quantified transition relations and initial state descriptions.
Conclusions and Future Work

We have given the decidability of the restriction to safety properties of the model-checking problem modulo locally finite and compatible theories;

Three main lines of future work:

1. how to exploit SMT solvers to solve model-checking problems (i.e., find suitable heuristics to efficiently explore the safety graph);
2. find decidability results for model checking of arbitrary temporal properties and modulo richer background theories [Demri et al. 2006];
3. handle universally quantified transition relations and initial state descriptions.
Conclusions and Future Work

- We have given the decidability of the restriction to safety properties of the model-checking problem modulo locally finite and compatible theories;

- Three main lines of future work:
  1. how to exploit SMT solvers to solve model-checking problems (i.e., find suitable heuristics to efficiently explore the safety graph);
  2. find decidability results for model checking of arbitrary temporal properties and modulo richer background theories [Demri et al. 2006];
  3. handle universally quantified transition relations and initial state descriptions.

